무선인터넷 플랫폼을 위한 네트워크 API 설계 및 구현

박충범*, 김연수, 연대진, 유용덕, 최 훈
충남대학교 컴퓨터공학과
e-mail : {cbpark,kimys,djyoun,ydyu,hchoi }@ce.cnu.ac.kr

Design and Implementation of Network API on Wireless Internet Platform

Choong-Bum* Park, Youn-Soo Kim, Dae-Jin Youn, Yong-Duck Yoo, Hoon Choi
Dept. of Computer Engineering, Chungnam National University

요 약

현재 국내 무선인터넷 표준 플랫폼 WIPI 가 제정됨에 따라 응용프로그램 개발자들을 지원하기 위한 WIPI 에뮬레이터의 필요성이 대두되었다. 이 논문에서는 본 연구팀에서 개발한 WIPI 에뮬레이터의 네트워크 기능을 지원하기 위한 socket API 및 HTTP API 로 구성되는 네트워크 API 의 설계/구현 사항을 제시하고, 플랫폼 검증 툴킷과 응용프로그램을 이용한 검증과정을 통하여 네트워크 API 의 적합성 검증 및 발전 방안을 제시한다.

1. 서론

과거 국내 무선인터넷 시장은 이동통신사마다 서로 다른 플랫폼 사용으로 인하여 플랫폼 호환성이나 컨텐츠 호환성이 제공되지 못하였다. 이러한 불합리성을 극복하고자 이동통신 3 사의 플랫폼을 하나의 규격으로 통일하는 무선인터넷 표준 플랫폼 WIPI(Wireless Internet Platform for Interoperability)가 제정되었으며 2004 년 9 월 WIPI 버전 2.0.1 이 발표되었다[1][2][3].

본 연구팀에서는 WIPI 플랫폼 규격을 따르는 컨텐츠 개발의 활성화를 위하여 한국전자통신연구원의 지원 하에 WIPI 에뮬레이터를 개발하였고, WIPI Clet 개발자를 지원하기 위한 WIPI C API 를 구현하였다[4][5].

특히 WIPI 플랫폼은 네트워크를 통하여 컨텐츠의 다운로드뿐만 아니라 WIPI 플랫폼 자체의 업데이트 기능도 지원하므로 강력한 네트워크 기능을 지원하는 네트워크 API 의 개발이 필요하다[3][6].

네트워크 API 는 TCP/IP 인터넷 통신에 관련된 모듈로서 인터넷 접근, TCP/UDP 소켓, HTTP 연결에 관련된 API 들이 포함된다.

* 본 연구는 정보통신부의 선도기반기술사업의 지원으로 수행되었음

이 논문에서는 본 연구팀이 개발한 네트워크 API 의 설계/구현 및 검증과정에 대하여 기술한다.

2 장에서는 네트워크 API 의 설계/구현의 사항을 제시하고, 3 장에서는 네트워크 API 의 동작원리를 기술한다. 4 장에서는 구현한 네트워크 API 의 검증과정에 대하여 기술하고, 5 장에서는 결론 및 개선 방안에 대하여 기술한다.

2. 네트워크 API 설계 및 구현

2-1. 네트워크 API 의 분류

네트워크 API 는 그 기능에 따라 socket API 와 HTTP API, 구조체의 초기화와 이벤트처리를 위한 보조 API 로 분류된다.

socket API 는 TCP 와 UDP 소켓 통신을 지원하기 위한 API 로서 HAL 을 통하여 윈도우의 Win32 socket 라이브러리를 이용할 수 있는 인터페이스 역할을 한다. 여기서 HAL(Handset Adaptation Layer)은 플랫폼의 이식에 있어 하드웨어의 독립성을 지원하기 위한 계층으로서 WIPI 에뮬레이터에서는 WIPI 플랫폼과 각종 라이브러리의 사이에 위치하게 된다.

HTTP API 는 HTTP(Hypertext Transfer Protocol)을 지

원하기 위한 API 이다. HTTP 는 응용계층의 프로토콜이며, HTTP API 는 socket API 를 이용하여 구현하였다. 보조 API 는 socket API 와 HTTP API 의 사용을 지원하기 위해 정의한 API 이다. 세부적인 분류는 <표 1>과 같다.

<표 1> 네트워크 API 의 분류

네트워크 API	**socket API**	socket 통신 준비 API
		TCP 통신 관련 API
		UDP 통신 관련 API
	HTTP API	HTTP 요청 준비 API
		HTTP 요청 API
		HTTP 응답 관련 API
		HTTP 해제 API
	보조 API	구조체 초기화 API
		이벤트 처리 API

2-2. 네트워크 API 설계

WIPI 플랫폼은 다중 프로그램의 수행을 지원하므로 네트워크를 사용하는 중에도 걸려오는 전화나 SMS 메시지를 수신할 수 있어야 한다. 이에 따라 모든 네트워크 API 는 넌블로킹(nonblocking)으로 설계하였다. 넌블로킹 API 는 호출과 동시에 결과값을 반환하지 못하는 경우도 발생하게 되며, 이 경우에는 아직 수행 중임을 의미하는 실패값 M_E_WOULDBLOCK 을 반환한다. 네트워크 API 의 실제 수행결과는 이벤트로 플랫폼에 전달되고, 플랫폼으로 전달된 이벤트를 받기 위한 콜백 함수(callback funtion)들을 설계하였다.

socket API 와 HTTP API 의 상태정보 저장을 위해 구조체 SOCKETINFO 와 httpInfo 를 정의하였다. 각 구조체의 내용은 [그림 1]과 같다.

```
typedef struct _SOCKETINFO {
        M_Int32 socketFD;
        NETSOCKCONNECTCB connectCB;
        NETSOCKWRITECB writeCB;
        NETSOCKREADCB readCB;
        void *cparam;
        void *wparam;
        void *rparam;
} SOCKETINFO;
```

```
typedef struct _httpInfo{
        M_Int32 fd;
        M_Int32 appID;
        M_Int32 connect;
        M_Int32 socketID;
        proxyInfo proxy;
        M_Byte *tempBuf[MAX_HTTP_PROPERTY];
        M_Int32 tempBufID[MAX_HTTP_PROPERTY];
        M_Byte *readBuf;
        M_Int32 readBufID;
}httpInfo;
```

[그림 1] 상태정보 저장을 위한 구조체

SOCKETINFO 구조체

- **socketFD:** 소켓 식별자
- **connectCB:** 소켓 연결 수행 시 발생하는 이벤트를 받기 위한 콜백 함수
- **writeCB:** 소켓 쓰기 수행 시 발생하는 이벤트를 받기 위한 콜백 함수
- **readCB:** 소켓 읽기 수행 시 발생하는 이벤트를 받기 위한 콜백 함수
- **cparam, wparam, rparam:** 각각의 콜백 함수에 전달할 매개변수

httpInfo 구조체

- fd: HTTP 식별자
- **appID:** httpInfo 를 사용하는 응용프로그램 ID, httpInfo 의 접근 권한
- **connect:** httpInfo 연결 상태정보
- **socketID:** HTTP 요청/응답에 사용되는 소켓 식별자
- **tempBuf[]:** HTTP 요청 메시지(request message) 구성을 위한 헤더의 속성
- **readBuf:** HTTP 응답 메시지(response message)를 저장하는 버퍼

2-3 네트워크 API 구현

네트워크 API 는 스펙에서 규정한 15 개의 socket API 와 15 개의 HTTP API 외에 네트워크 사용에 이용되는 구조체의 초기화 함수인 netInit()와 플랫폼으로부터 이벤트를 받기 위한 handleNetworkEvent() 함수를 보조 API 로 정의하고 설계 및 구현하였다.

네트워크 API 의 핵심이 되는 MC_netHttpConnect() 함수는 [그림 2]와 같이 구현하였다.

[그림 2] MC_netHttpConnect()의 설계

MC_netHttpConnect() 함수는 서버와 HTTP 연결을 시도하는 함수로서 서버로 전달할 메시지를 구성하고 전달하며 서버로부터 받은 응답을 저장하고 해석하는 기능을 한다. MC_netHttpConnect() 함수를 호출할 때 식별자 fd 와 콜백 함수 cb, 콜백 함수의 파라미터 param 을 파라미터로 전달한다.

① appID 를 검사하여 사용 권한이 없는 응용프로그램의 접근을 방지한다.
② fd 값의 유효성을 검사하여 적합한 HTTP 연결을 시도한다.
③ 메시지를 전달할 소켓을 생성을 위해 MC_netSocket() 함수를 호출한다.
④ httpInfo 에 저장한 tempBuf[0] 값을 참고하여 접속할 서버의 IP 주소와 port 번호를 구한다.
⑤ tempBuf[]에 저장한 값들을 참고하여 서버로 전송할 메시지를 구성한다.
⑥ 생성한 소켓을 통한 서버와의 연결 시도를 위해 MC_netSocketConnect() 함수를 호출한다.
⑦ 서버로 메시지를 전달하기 위해 MC_netSocketWrite() 함수를 호출한다.
⑧ 서버로부터 받은 응답을 Readbuf 에 저장하기 위해 MC_netSocketRead() 함수를 호출한다.
⑨ readBuf 에 저장된 메시지를 해석한다.
⑩ MC_netHttpConnect() 함수 호출 시 등록한 콜백 함수를 호출하여 플랫폼으로 전달되는 이벤트를 얻어온다.

⑥⑦⑧단계에서 호출되는 socket API 는 반환값으로 M_E_WOULDBLOCK 을 발생할 수 있는 넌블로킹 함수이다. 각 함수가 호출되면 구조체 SOCKETINFO 에 등록된 각각의 콜백 함수가 호출되며 플랫폼으로부터 각각의 socket API 의 호출 결과를 이벤트로 얻어오게 된다. MC_netHttpInfo() 함수 역시 M_E_WOULDBLOCK 을 발생할 수 있는 함수이므로 마지막 수행 단계로 호출 시 파라미터로 입력받은 NETHTTPCB 타입의 콜백 함수 cb 를 호출하게 된다.

3. 네트워크 API 동작원리

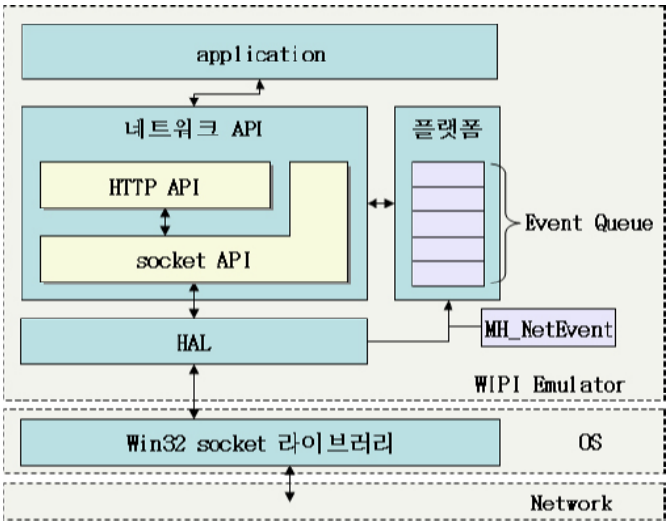
본 연구팀이 개발한 에뮬레이터는 윈도우 환경에서 실행되므로 실제 단말기가 무선인터넷을 통하여 통신이 이루어지는 것과 달리 에뮬레이터를 실행하는 컴퓨터 네트워크를 통하여 통신이 이루어진다. 컴퓨터 네트워크를 이용하기 위해서 에뮬레이터가 윈도우에서 제공하는 Win32 socket 라이브러리를 이용하도록 구현하였다.

에뮬레이터에서 실행되는 응용프로그램은 통신을 위해 네트워크 API 를 호출하게 된다. 사용 목적에 따라 HTTP 연결을 위해서는 HTTP API 를 통하여 socket API 를 사용하고, TCP/UDP 소켓 통신을 위해서는 socket API 를 직접 이용하게 된다.

네트워크 API 는 넌블로킹 함수이므로 네트워크 API 호출에 따른 실제 성공/실패 결과는 네트워크 관련 이벤트 MH_NETWORK_EVENT 를 통하여 플랫폼에 전달되고, 플랫폼내의 이벤트 큐(Event Queue)에 저장된다.

플랫폼 이벤트 큐에 MH_NETWORK_EVENT 가 전달되면 MH_NETWORK_EVENT 를 파라미터로 참조하여 handleNetworkEvent() 함수를 호출한다.

handleNetworkEvent() 함수는 파라미터로 전달받은 MH_NETWORK_EVENT 를 내용을 분석하여 해당 이벤트를 발생시킨 네트워크 API 에 관련된 콜백 함수를 실행시킨다.



[그림 3] 네트워크 API 동작원리

4. WIPI 네트워크 API 검증

설계 및 구현한 네트워크 API 의 적합성을 검증하기 위해 PCT 를 통한 검증과 테스트용 Clet 을 통한 검증을 수행하였다.

4-1. PCT 를 통한 네트워크 API 검증

PCT(Platform Certification Toolkit)는 EXEMobile㈜에서 개발한 WIPI 플랫폼 검증 툴킷이다.

PCT 를 통한 API 검증은 각각의 API 를 임의의 파라미터를 이용하여 호출하고, 전달한 파라미터에 따라 적합한 성공/실패 값을 반환하는지 검사하는 형식으로 실행하였다.

네트워크 API 의 검증은 모두 33 가지 항목으로 이루어지며 주요 항목은 다음과 같다.

- **MC_netConnect100x:** 임의의 파라미터들로 인터넷 설정을 시도하고, 적합한 결과값의 반환 여부 확인
- **MC_Socket100x:** 임의의 파라미터들로 소켓 생성을 시도하고, 적합한 결과값의 반환여부 확인
- **MC_netSocketclose100x:** 임의의 파라미터들로 소켓 소멸을 시도하고, 적합한 결과값의 반환 여부 확인

- **MC_netSocketRcvFrom100x:** 임의의 파라미터들로 소켓 전송을 시도하고, 적합한 결과값의 반환여부 확인
- **MC_netSocketRead100x:** 임의의 파라미터들로 소켓 수신을 시도하고, 적합한 결과값의 반환여부 확인
- **MC_netHttpOpen100x:** 임의의 파라미터들로 HTTP 연결을 시도하고, 적합한 결과값의 반환여부 확인
- **MC_netHttpConnect100x:** 임의의 파라미터들로 HTTP 메시지 송/수신을 시도하고, 적합한 결과값의 반환여부 확인

[그림 4] PCT 를 통한 네트워크 API 검증

PCT 를 통한 검증에서 33 가지의 네트워크 검증항목을 모두 통과하였다.

4-2. 테스트용 Clet 을 통한 네트워크 API 의 검증

한국전자통신연구원에서 제공한 플랫폼 검증 응용프로그램인 테스트용 Clet 을 이용하여 구현한 네트워크 API 의 적합성을 검증하였다.

테스트용 Clet 은 네트워크 API 의 HTTP API 와 socket API 를 이용하여 웹 서버에 접속하여 미리 등록된 bmp, png, gif 포맷의 이미지 파일을 다운로드하는 기능과 다운로드한 이미지를 에뮬레이터 스크린에 표시하는 기능을 가지고 있다.

[그림 5] 테스트용 Clet 을 통한 네트워크 API 검증

테스트용 Clet 의 실행을 통한 네트워크 API 검증 과정에서 이미지가 다운로드되어 표시되는 것을 확인함으로써 네트워크 API 의 적합성을 검증하였다.

5. 결론

본 연구팀은 설계 및 구현한 네트워크 API 의 PCT 를 통한 검증과 테스트용 Clet 을 통한 검증을 통해 적합성을 확인하였다.

WIPI 에뮬레이터는 일반 단말기와 달리 무선 네트워크가 아닌 사용자의 컴퓨터 네트워크를 이용하여 통신이 이루어지게 되고, 이를 위하여 Win32 socket 라이브러리를 이용하게 되므로 윈도우에 종속적이게 되는 문제가 있다.

하지만 일반 단말기의 무선 네트워크의 통신 속도가 수백 kbps 에 머무는데 비해 컴퓨터 네트워크를 이용한 통신은 수 Mbps 의 통신 속도를 지원하므로 WIPI 에뮬레이터가 사용자가 아닌 개발자의 지원을 위한 프로그램임을 감안할 때 장점으로 작용될 수 있다.

향후 보다 다양한 개발자들의 요구를 충족시키기 위해서 윈도우에 종속적인 일부 네트워크 API 를 운영체제에 독립적으로 실행할 수 있도록 설계 및 구현하여 윈도우 외에 운영체제를 사용하는 개발자도 WIPI 에뮬레이터를 사용할 수 있도록 지원할 계획이다.

참고문헌

[1] "한국무선인터넷표준화포럼" www.kwisforum.org, 2004.9.9
[2] 모바일 표준 플랫폼 WIPI 1.2.1, KWISFS.K-05-001R3
[3] 모바일 표준 플랫폼 WIPI 2.0, KWISFS.K-05-002
[4] "윈도우용 Clet 에뮬레이터 개발" , ETRI 연구보고서, 충남대학교, 2004.8
[5] "휴대정보단말기용 웹 서비스 클라이언트 및 WIPI SDK", ETRI 연구보고서, 충남대학교, 2003.12
[6] 유용덕, 김연수, 최훈, "동적 Upgrade 기능을 구비한 WIPI 개발환경 설계", 한국통신학회 추계 종합학술발표회 논문초록집, vol. 28, p.343, 2003.12
[7] "MOBILEJAVA Developer Community", www.mobilejava.co.kr, 2004.9.9
[8] "HTTP - Hypertext Transfer Protocol Overview", www.w3.org/Protocols/, 2004,10,2
[9] Michael J. Donalhoo, Kenneth L. Calvert, 박준철 역 "TCP/IP 소켓프로그래밍 C version", 사이텍미디어 2001.10